

Tutorial

# omicR

Windows, Mac and Linux

Berenice Talamantes Becerra  
8-26-2020

<b><i>Figures</i></b>	<b>2</b>
<b><i>Introduction</i></b>	<b>3</b>
<b><i>OmicR with graphical user interface</i></b>	<b>4</b>
<b>Windows</b>	<b>4</b>
<b>MAC</b>	<b>5</b>
<b>Running “omicR” with graphical user interface</b>	<b>6</b>
2.     Download genomes from the NCBI.	9
3.     Create NCBI database for BLAST+.	9
4.     BLAST and filtering.	10
Description of files produced.	11
5.     Additional filtering.	12
<b><i>omicR for R Studio</i></b>	<b>13</b>
<b>Running “omicR” in R studio</b>	<b>13</b>
Create FASTA files and input files for BLAST / filtering.	14
Download genomes from the NCBI.	16
Create BLAST+ database.	18
BLAST and filtering.	19
Description of files produced.	21
Running BLAST and filtering script without the file with Unique ID.	23
Additional filtering.	23
Common errors	25
<b><i>omicR for HPC computers</i></b>	<b>26</b>
1.     Create FASTA files and input files for BLAST / filtering.	26
2.     Download genomes from the NCBI.	28
3.     Create NCBI database for BLAST+.	30
4.     BLAST and filtering.	31
Description of files produced.	33
5.     Additional filtering.	34

## Figures

Figure 1 omicR logo.....	4
Figure 2 Sample Data of E. faecium .....	6
Figure 3 First windows of omicR GUI.....	7
Figure 4 Create Fasta files and input files for BLAST using GUI. ....	7
Figure 5 Display of how rows and headers should be selected.....	8
Figure 6 Output files generated from script "Create Fasta files". .....	8
Figure 7 Download genome entries from NCBI. ....	9
Figure 8 Create NCBI genome database for BLAST+.....	9
Figure 9 Example of files created after creating BLAST database. ....	10
Figure 10 BLAST and filtering example. ....	10
Figure 11 Example of files produced after running the BLAST and filtering script.....	11
Figure 12 Example of CSV input file. ....	13
Figure 13 mkfastafile.R script before adding details. ....	14
Figure 14 mkfastafile.R script after adding details. ....	15
Figure 15 mkfastafile.R script after selecting all the code.....	15
Figure 16 mkfastafile.R script after running the code. ....	15
Figure 17 Example of fasta file and file with Unique ID.....	16
Figure 18 Example of downloadGenomes.R script. ....	16
Figure 19 Example of downloadGenomes.R script after adding parameters.....	17
Figure 20 Example of downloadGenomes.R script after running the code.....	17
Figure 21 Example of window showing "makeDatabase.R" .....	18
Figure 22 Example of script "makeDatabase.R" after filling parameters. ....	18
Figure 23 Example of script "makeDatabase.R" after running the code. ....	19
Figure 24 Example of files created with the code "makeDatabase.R" .	19
Figure 25 Example of window showing BLASTnFilter.R script. ....	20
Figure 26 Example of script BLASTnFilter.R after filling parameters. ....	20
Figure 27 Example of script BLASTnFilter.R after running the code. ....	21
Figure 28 Example of files created by BLASTnFilter.R script.....	21
Figure 29 Example of script BLASTnFilter.R after running the code without the file with UniqueID... <td>23</td>	23
Figure 30 Example output files generated after running the script BLASTnFilter.R without the file with UniqueID. ....	23
Figure 31 Example of filter.R script.....	24
Figure 32 Example of "filter.R script" after filling parameters. ....	24
Figure 33 Example of "filter.R script" after running the code. ....	25
Figure 34 Display of how rows and headers should be selected.....	27

## Introduction

omicR creates fasta files, downloads genomes from NCBI using the refseq number, creates databases to run BLAST+, runs BLAST+ and filters these results to obtain the best match per sequence.

These scripts can be used to run BLAST alignment of short-read (DArTseq data) and long-read sequences (Illumina, PacBio... etc). You can use reference genomes from NCBI, genomes from your private collection, contigs, scaffolds or any other genetic sequence that you would like to use as reference.



You can skip this tutorial and watch the tutorial video in YouTube

omicR with Graphical User Interface (~20 min)

<https://youtu.be/pdMio2vj-FM>

omicR in R Studio (~20 min)

<https://youtu.be/2dEqOBjcvM8>

# OmicR with graphical user interface

## Windows

### Requirements:

BLAST+ latest version: <https://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/>

### Installation:

You can download these scripts from Github:

[https://github.com/BTalamantesBecerra/omicR\\_for\\_Windows](https://github.com/BTalamantesBecerra/omicR_for_Windows)

There are 2 options to run omicR in windows.

- 1) **Option 1.** Download the zip directory “omicR.zip”. Unzip this directory and double click the executable file “omicR.exe” with the image of the green parrot. This will open a window where you can run the scripts. You do not need to install anything as everything is compiled into this file and you can start running your analysis.

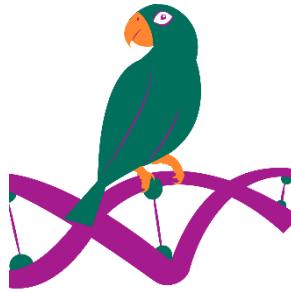


Figure 1 omicR logo.

- 2) **Option2.** If you cannot open the executable file, you may need to run the script directly through Python. For this you need to install the following:
  - a. Python V3 or latest: <https://www.python.org/downloads/>
  - b. Biopython: <https://biopython.org/>
  - c. omicR: <https://github.com/BTalamantesBecerra/omicR> Download the following files from GitHub: “omicR.py” and “Currito.ico” and save them in your python working directory.

-Open the script “omicR.py” in Python and run it.

-This will open a window where you can run the scripts.

As general practice avoid installing your software in directories such as “C://Program files/” as the space between words will cause problems.

Remember to add all the software you install into your System Variables Path.

## MAC

Requirements:

- a. Python V3 or latest: <https://www.python.org/downloads/>
- b. Biopython <https://biopython.org/>
- c. BLAST+ latest version: <https://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/>
- d. omicR: <https://github.com/BTalamantesBecerra/omicR> Download the following files “omicR.py” and “Currito.ico” from GitHub and save them in your Python working directory.

-Open the script “omicR.py” in Python and run it.

-This will open a window where you can run the scripts.

Remember to add all the software you install into your System Variables Path.

## Running “omicR” with graphical user interface

Before you run a test, download the Sample data. The csv file is called “SampleData\_Enterococcus\_faecium.csv”.

**Only CSV files are accepted as input for these scripts.**

SeqIndex	ClusterIdx	ClusterSize	Tag	TrimmedSLength	LowComL	Chrom_d	ChromPos	AlnCnt_da	AlnValue	AdapterPc	NumPrese	HighestCo	CountSum	AvgNonZe	E6	H7	D3	
1	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*			
2	*	*	*	*	*	*	*	*	*	*	*	*	*	*	900716298002	900716298002	90	
3	*	*	*	*	*	*	*	*	*	*	*	*	*	*	2	2		
4	*	*	*	*	*	*	*	*	*	*	*	*	*	*	152_2014_ENT163_2015_ENT12			
5	*	*	*	*	*	*	*	*	*	*	*	*	*	*	161348	151821		
6	SeqIndex	ClusterIdx	ClusterSize	Tag	TrimmedSLength	LowComL	Chrom_d	ChromPos	AlnCnt_da	AlnValue	AdapterPc	NumPrese	HighestCo	CountSum	AvgNonZe	999068	999079	
7	89184	20963	408	TGCAGAA/TGCAGAA	49	0	0	0	999	49	43	51484	596359	13868.81	16911	0		
8	75556	20878	111	TGCAGCG/TGCAGCG	46	0	0	0	999	46	14	12437	107456	7675.429	0	8728		
9	73901	20862	96	TGCAGTA/TGCAGTA	41	0	0	0	999	41	16	12381	90809	5675.563	0	6452		
10	79622	20911	137	TGCAGTT/TGCAGTT	40	0	0	0	999	40	82	3987	215378	2626.561	3001	2180		
11	78436	20902	127	TGCAGCT/TGCAGCT	43	0	0	0	999	43	82	3163	180355	2199.451	2475	2096		
12	78175	20900	126	TGCAGCA/TGCAGCA	46	0	0	0	999	46	80	3093	162945	2036.813	2461	2068		
13	79343	20909	134	TGCAGAT/TGCAGAT	52	0	0	0	999	52	81	2969	138818	1713.802	1827	1779		
14	76464	20886	118	TGCAGAA/TGCAGAA	43	0	0	0	999	43	83	4764	152191	1833.627	1943	1608		
15	79759	20912	137	TGCAGTT/TGCAGTT	52	0	0	0	999	52	79	2560	132856	1681.722	1792	2310		
16	80025	20914	139	TGCAGGA/TGCAGGA	53	0	0	0	999	53	79	2736	133532	1690.278	2040	1657		
17	79492	20910	136	TGCAGCA/TGCAGCA	45	0	0	0	999	45	80	2759	138077	1725.963	2274	1664		
18	80161	20915	141	TGCAGAG/TGCAGAG	51	0	0	0	999	51	79	3700	133197	1866.038	1728	1621		
19	76107	20883	117	TGCAGTA/TGCAGTA	46	0	0	0	999	46	80	2686	131079	1638.488	1814	1622		
20	82781	20933	156	TGCAGTT/TGCAGTT	69	0	0	0	999	0	80	2335	114989	1437.363	1580	1466		

Figure 2 Sample Data of *E. faecium*

Open or run the “omicR” executable.

The window should look like this. As you can see, each button runs a different script.

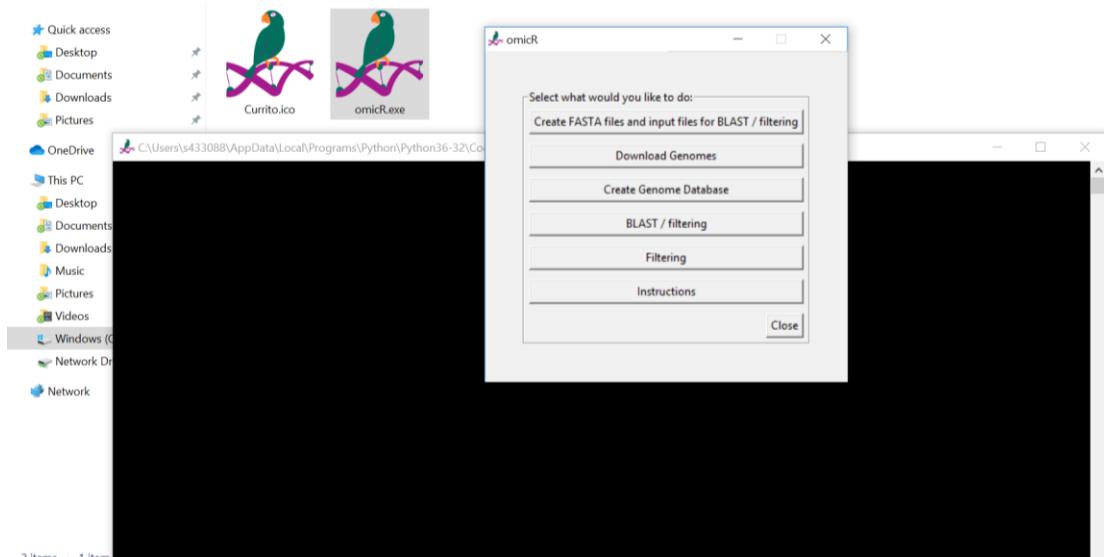


Figure 3 First windows of omicR GUI.

# STEPS

## 1. Create FASTA files and input files for BLAST / filtering.

This will open another window. Select the paths to the Sample Data file provided and select parameters as required.

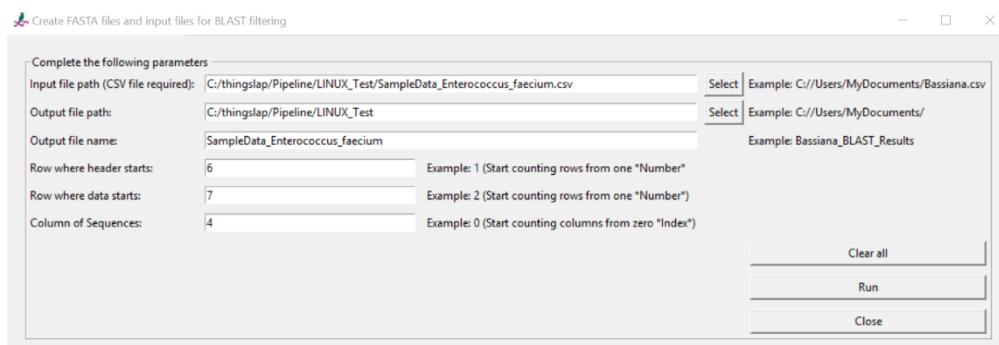


Figure 4 Create Fasta files and input files for BLAST using GUI.

The row where header starts is row 6 and the row where data starts is 7. And in this case, for this file we need the Trimmed Sequences column, it is in column 4. For columns you need to start counting from 0.

Sequence of interest: Column 4

	A	B	C	D	E	F	G	H	I	J	K
1	*	*	*	*	*	*	*	*	*	*	*
2	*	*	*	*	*	*	*	*	*	*	*
3	*	*	*	*	*	*	*	*	*	*	*
4	*	*	*	*	*	*	*	*	*	*	*
5	*	*	*	*	*	*	*	*	*	*	*
Header: Row 6	SeqIndex	ClusterIdx	ClusterSiz	Tag	TrimmedSLength	LowComlChrom	ChromPos	ChromPos	AlnCnt	AlnValue	
Data: Row 7	7	89184	20963	408	TGCAGAA/TGCAGAA	49	0	0	0	0	999
	8	75556	20878	111	TGCAGCG/TGCAGCG	46	0	0	0	0	999
	9	73901	20862	96	TGCAGTA/TGCAGTA	41	0	0	0	0	999
	10	79622	20911	137	TGCAGTT/TGCAGTT	40	0	0	0	0	999
	11	78436	20902	127	TGCAGCT/TGCAGCT	43	0	0	0	0	999
	12	78175	20900	126	TGCAGCA/TGCAGCA	46	0	0	0	0	999
	13	79343	20909	134	TGCAGAT/TGCAGAT	52	0	0	0	0	999

Figure 5 Display of how rows and headers should be selected.

### REMEMBER TO CHECK THE HEADER AND DATA OF YOUR FILE BEFORE RUNNING ANY SCRIPT

This step should be completed in less than 1 minute.

This will create 2 files, one Fasta file and one copy of your original file including an extra column containing a Unique ID. These files are required for the following steps.

Fasta file											
File	Home	Insert	Page Layout	Formulas	Data	Review	View				
AutoSave	Off	Save	Undo	Redo	Font	Font	Font	Font	Font	Font	Font
1	>1										
2	TGCAGAGAAGTACGAAGAGAACAGAACACTTCAGCCTGAAACACCG										
3	>2										
4	TGCAGCGGCCATCATACGGGATAACGACTGTATGACGTGAAACCG										
5	>3										
6	TGCAGTACGGAATTCCTGATTATCAGGAACTCGAGCGCC										
7	>4										
8	TGCAGTTGCTGTCCTGGCACCATATTTCGCGAAGTCG										
9	>5										
10	TGCAGCTGCATTGGCTCGATTACTTGATGCAAGAACATCCG										
11	>6										
12	TGCAGCATCGCTTGAAAGACTAGGCGTTACGTATTAGAACCG										
13	>7										
14	TGCAGATGATATCCGTACCTAGCTGAACGATTAGAACGAAACACTCG										
15	>8										
16	TGCAGAACGCAATCATATATTGGCTAACGATTGTGCCCC										
17	>9										
18	TGCAGTTCTGGTAAATTCTCTAGCATCACCAAGAACGAGAACCG										
19	>10										
20	TGCAGGAGCTGTTTGAGTTACAGAACCGAGAACGGAGAACCGCTACAAACCG										

File with FastaFileID											
File	Home	Insert	Page Layout	Formulas	Data	Review	View				
AutoSave	Off	Save	Undo	Redo	Font	Font	Font	Font	Font	Font	Font
A1											
	FastaFileID										
1	FastaFileID	SeqIndex	ClusterIdx	ClusterSiz	Tag	TrimmedSequence					
2	1	89184	20963	408	TGCAGAAAGTACGAAGAGAAC						
3	2	75556	20878	111	TGCAGCG/TGCAGGCC						
4	3	73901	20862	96	TGCAGTA/TGCAGTA						
5	4	79622	20911	137	TGCAGTT/TGCAGTT						
6	5	78436	20902	127	TGCAGCT/TGCAGCT						
7	6	78175	20900	126	TGCAGCA/TGCAGCA						
8	7	79343	20909	134	TGCAGAT/TGCAGAT						
9	8	76464	20886	118	TGCAGAA/TGCAGAAC						
10	9	79759	20912	137	TGCAGTT/TGCAGTT						
11	10	80025	20914	139	TGCAGGA/TGCAGGAG						
12	11	79492	20910	136	TGCAGCA/TGCAGCA						
13	12	80161	20915	141	TGCAGAG/TGCAGAG						
14	13	76107	20883	117	TGCAGTA/TGCAGTA						
15	14	82781	20933	156	TGCAGTT/TGCAGTT						
16	15	75759	20880	112	TGCAGCA/TGCAGCAA						
17	16	85903	20950	214	TGCAGTT/TGCAGTT						
18	17	83279	20936	163	TGCAGTA/TGCAGTA						
19	18	83781	20939	179	TGCAGCA/TGCAGCAC						
20	19	83415	20937	171	TGCAGAT/TGCAGAT						

Figure 6 Output files generated from script "Create Fasta files".

## 2. Download genomes from the NCBI.

For this example, we need the genome and plasmid of *E. faecium* ([https://www.ncbi.nlm.nih.gov/assembly/GCF\\_010120755.1](https://www.ncbi.nlm.nih.gov/assembly/GCF_010120755.1)). The RefSeq numbers needed for this tutorial are: NZ\_CP039729.1, NZ\_CP039730.1

To run the script, write your email, select the RefSeq accession numbers, select the output path and name for the genome.

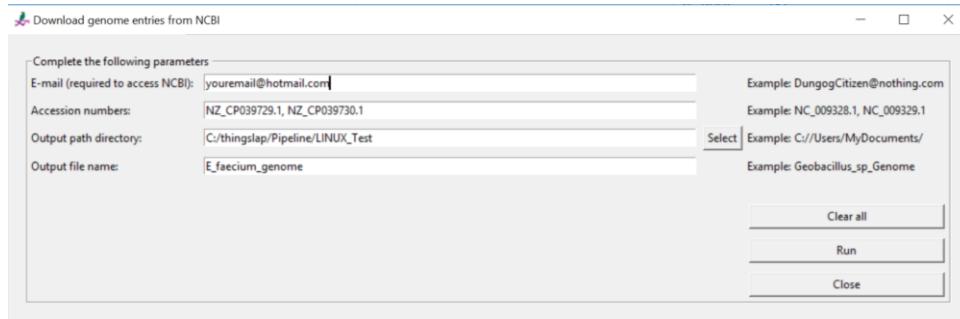


Figure 7 Download genome entries from NCBI.

This will create a directory with the name given to the genome.

Please note that this option is suitable for small genomes or small chromosomes. To download larger genomes, it is recommended to use the internet browser option. This example of *E. faecium* should download in less than 5 minutes. If you use this method to download the Chicken genome it can take up to 5 hours.

## 3. Create NCBI database for BLAST+.

To create the database for BLAST, select the path to the "bin" directory where BLAST+ was installed.

Select the location of the genome/sequences/scaffolds/contigs or whatever you would like to use as reference to BLAST, then select the type of database. For this example we are working with nucleotides, so we select "nucl".

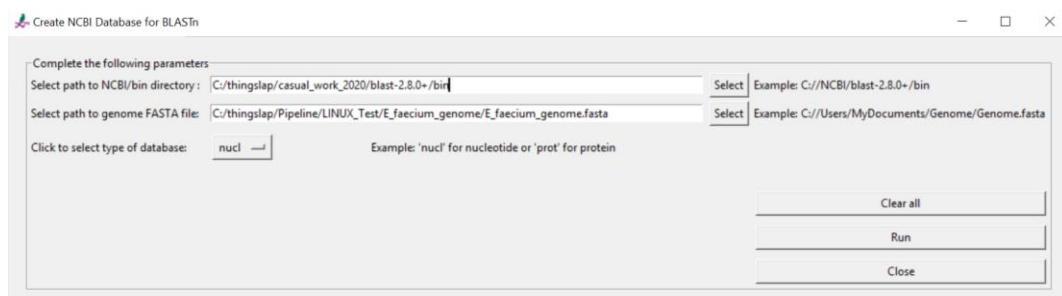


Figure 8 Create NCBI genome database for BLAST+.

The script will create 3 files in the same location as your database, with the same name of the reference and the terminations “.nhr”, “nin”, “nsq”.

Name	Date modified	Type	Size
E_faecium_genome.fasta	25/08/2020 11:57 AM	FASTA File	2,769 KB
E_faecium_genome.fasta.nhr	25/08/2020 12:08 PM	NHR File	1 KB
E_faecium_genome.fasta.nin	25/08/2020 12:08 PM	NIN File	1 KB
E_faecium_genome.fasta.nsq	25/08/2020 12:08 PM	NSQ File	692 KB

Figure 9 Example of files created after creating BLAST database.

#### 4. BLAST and filtering.

To run the BLAST analysis, you need to select the path to the bin directory where you installed BLAST+, then select your path to the database created in the previous step, the output path, the output file name, the output path to the file with Unique ID and BLAST parameters.

##### NOTES:

- If you only used a fasta file as input, and you do not have the file with Unique ID, you can still run this script.
- If you are running a BLAST alignment of similar sequences, for example Turtle Genome Vs Turtle Sequences, the recommended parameters are: Word Size 11, Percentage identity 70, Number of threads 4, Output format 6, Percentage Overlap 0.8, bitscore 50.
- If you are running a BLAST of highly dissimilar sequences because you are probably looking for sex linked hits in a distantly related species, and you are aligning sequences of Chicken Genome Vs Bassiana, use a Percentage overlap of 0.01, Bitscore of 30 and tick the option of “Discontinuous Mega BLAST”

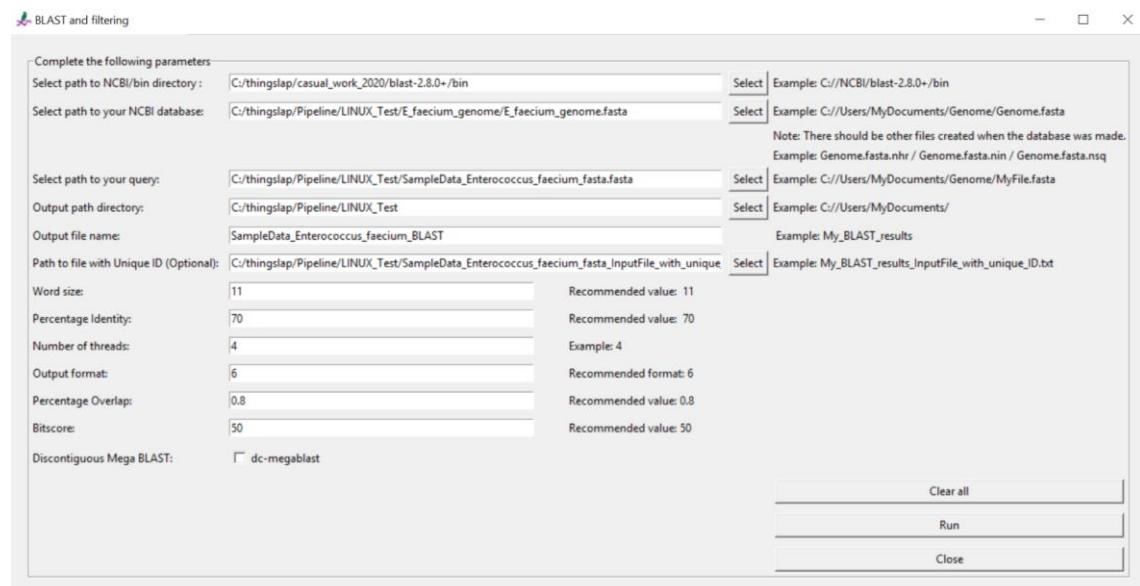


Figure 10 BLAST and filtering example.

This step takes less than 5 minutes. It will produce 5 files, or 3 files if you didn't provide the File with UniqueID.

SampleData_Enterococcus_faecium_BLAST_all_sequences_with_and_without_hits.txt	24/08/2020 5:53 PM	Text Document	425 KB
SampleData_Enterococcus_faecium_BLAST_only_sequences_with_hits.txt	24/08/2020 5:53 PM	Text Document	377 KB
SampleData_Enterococcus_faecium_BLAST_sorted.txt	24/08/2020 5:52 PM	Text Document	215 KB
SampleData_Enterococcus_faecium_BLAST_filtered.txt	24/08/2020 5:52 PM	Text Document	257 KB
SampleData_Enterococcus_faecium_BLASTBLAST.txt	24/08/2020 5:52 PM	Text Document	3,057 KB

Figure 11 Example of files produced after running the BLAST and filtering script.

### Description of files produced.

- **File 1. SampleData\_Enterococcus\_faecium\_BLASTBLAST.txt**

This is the raw BLAST output. This file does not contain any headers and it is not filtered.

- **File 2. SampleData\_Enterococcus\_faecium\_BLAST\_filtered.txt**

This file has headers and an extra column with the percentage overlap. Default filtering parameters for this tutorial are: Percentage Overlap >80%, bitscore >50, Percentage Identity>70. The percentage overlap can be modified according to the BLAST results expected.

This file may contain multiple hits per sequence.

The BLAST output is formatted as a table using output format 6, with columns defined in the following order: " qseqid sacc stitle qseq sseq nident mismatch pident length evalue bitscore qstart qend sstart send gapopen gaps qlen slen". These are:

- qseqid: query (e.g., unknown gene) sequence id
- sacc: Subject accession
- stitle: Subject Title
- qseq: Aligned part of query sequence
- sseq: Aligned part of subject sequence
- nident: Number of identical matches
- mismatch: number of mismatches
- pident: percentage of identical matches
- length: alignment length (sequence overlap)
- evalue: expect value
- bitscore: bit score
- qstart: start of alignment in query
- qend: end of alignment in query
- sstart: start of alignment in subject
- send: end of alignment in subject
- gapopen: number of gap openings
- gaps: Total number of gaps
- qlen: Query sequence length

- slen: Subject sequence length
- **File 3. SampleData\_Enterococcus\_faecium\_BLAST\_sorted.txt**  
 This file contains only one hit per sequence. The best match will be selected by considering the following values ranked in order. First considering the highest percentage identity, then the highest Percentage overlap, then the highest bitscore. Only one Query per sequence is kept based on these selection criteria.  
 If you did not provide the file with UniqueID, this filtered and sorted BLAST output file will be your final results.
- **File 4. SampleData\_Enterococcus\_faecium\_BLAST\_only\_sequences\_with\_hits.txt**  
 This file uses the UniqueID assigned to each sequence and writes the BLAST results back into the original file. This file only contains sequences that had a BLAST hit to something in the reference.
- **File 5. SampleData\_Enterococcus\_faecium\_BLAST\_all\_sequences\_with\_and\_without\_hits.txt**  
 This file uses the UniqueID assigned to each sequence and writes the BLAST results back into the original file. This file contains all sequences, including those with and without hits, written back into the original file.

## 5. Additional filtering.

If you would like to run additional filtering without re-running the BLAST, you can use the BLAST result obtained in the previous step as an input and filter again with different parameters. For example, using a higher or lower percentage overlap or bitscore.

Note: This script only takes input files with BLAST Tabular output format 6 with the ordered set of columns described in the previous step.

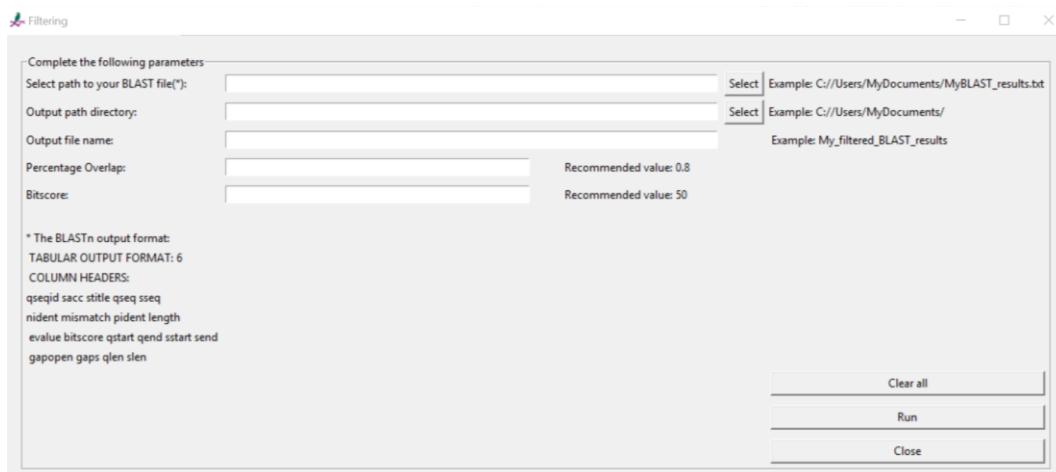


Figure Additional filtering window.

## omicR for R Studio

omicR for R studio runs using Python scripts through R Studio. You can download the R project in:  
[https://github.com/BTalamantesBecerra/omicR\\_for\\_RStudio](https://github.com/BTalamantesBecerra/omicR_for_RStudio)

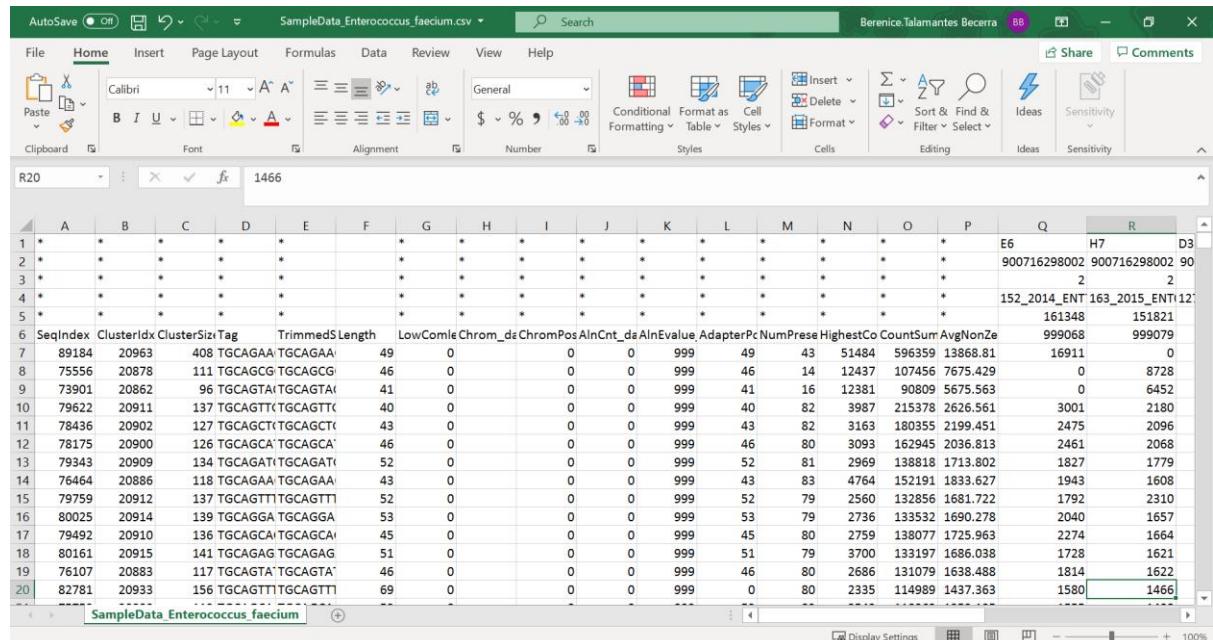
1) You need to install the following:

- a. Python V3 or latest: <https://www.python.org/downloads/>
- b. Biopython <https://biopython.org/>
- c. BLAST+ latest version: <https://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/>
- d. Download the omicR project. This should include at least the following:  
“omicR.Rproj”,  
“TestingPyCharm\_MKfasta.py”,  
“TestingPyCharm\_Downloading\_genomes.py”,  
“TestingPyCharm\_MakeDataBase.py”,  
“TestingPyCharm\_BLAST\_filtering\_and\_all.py”,  
“TestingPyCharm\_NCBI\_BLAST\_filtering.py”

## Running “omicR” in R studio

Before you run a test, download the Sample data. The csv file is called “SampleData\_Enterococcus\_faecium.csv”.

**Note: The pipeline only takes csv files as input.**



SeqIndex	ClusterIdx	ClusterSiz	Tag	TrimmedS	Length	LowComle	Chrom	ChromPos	AInCnt	daAIn	EValue	AdapterPc	NumPrese	HighestCo	CountSum	AvgNonZe	999068	999079
1	89184	20963	408	TGCAAGAA/TGCAAGAA	49	0	0	0	999	49	43	51484	596359	13868.81	16911	0		
2	75556	20878	111	TGCAGGG/TGCAGGG	46	0	0	0	999	46	14	12437	107456	7675.429	0	8728		
3	73901	20862	96	TGCAGTA/TGCAGTA	41	0	0	0	999	41	16	12381	90809	5675.563	0	6452		
4	79622	20911	137	TGCAGTT/TGCAGTT	40	0	0	0	999	40	82	3987	215378	2626.561	3001	2180		
5	78436	20902	127	TGCAGCT/TGCAGCT	43	0	0	0	999	43	82	3163	180355	2199.451	2475	2096		
6	78175	20900	126	TGCAGCA/TGCAGCA	46	0	0	0	999	46	80	3093	162945	2036.813	2461	2068		
7	79343	20909	134	TGCAGAT/TGCAGAT	52	0	0	0	999	52	81	2969	138818	1713.802	1827	1779		
8	76464	20886	118	TGCAAGAA/TGCAAGAA	43	0	0	0	999	43	83	4764	152191	1833.627	1943	1608		
9	79759	20912	137	TGCAGTT/TGCAGTT	52	0	0	0	999	52	79	2560	132856	1681.722	1792	2310		
10	80025	20914	139	TGCAGGA/TGCAGGA	53	0	0	0	999	53	79	2736	133532	1690.278	2040	1657		
11	79492	20910	136	TGCAGCA/TGCAGCA	45	0	0	0	999	45	80	2759	138077	1725.963	2274	1664		
12	80161	20915	141	TGCAGAG/TGCAGAG	51	0	0	0	999	51	79	3700	133197	1686.038	1728	1621		
13	76107	20883	117	TGCAGTA/TGCAGTA	46	0	0	0	999	46	80	2686	131079	1638.488	1814	1622		
14	82781	20933	156	TGCAGTT/TGCAGTT	69	0	0	0	999	0	80	2335	114989	1437.363	1580	1466		

Figure 12 Example of CSV input file.

# STEPS

## Create FASTA files and input files for BLAST / filtering.

- Select the script “mkfastafile.R”.
- Clean the global environment before running these scripts.

The screenshot shows the RStudio interface with the "omicR - RStudio" window open. In the top-left, there are tabs for "mkfastafile.R", "downloadGenomes.R", "makeDatabase.R", "BLASTnFilter.R", and "filter.R". The main area displays the "mkfastafile.R" script:

```
1 #MK FASTA
2 #Create FASTA FILES and FILE WITH UNIQUE ID for BLAST.
3 #PLEASE FILL THE FOLLOWING
4
5 python_exe <- ""
6 inputCSV_file_path <- ""
7 output_path <- ""
8 output_name <- ""
9 row_where_header_starts <- ""
10 row_where_data_starts <- ""
11 sequence_column <- "|"
12
13 ##### DO NOT MODIFY ANYTHING BELOW HERE#####
14 #####
15 #####
16 #####
17 #####
18 #####
19 #####
20 #####
21 #####
22
```

The "Global Environment" pane on the right shows that the "Environment" is empty. The "Files" pane shows a project structure under "ap > Pipeline > RPackage > omicR":

- Namespace
- omicR.Rproj
- R
- Report\_FreshWaterTurtle\_SNP\_m...
- TestingPyCharm\_BLAST\_filtering\_...
- TestingPyCharm\_Downloading.g...
- TestingPyCharm\_MakeDataBase.py
- TestingPyCharm\_MKfasta.py
- TestingPyCharm\_NCBI\_BLAST\_filt...

Figure 13 mkfastafile.R script before adding details.

- Now write the paths to your files as indicated. Do not modify anything below the sign “DO NOT MODIFY ANYTHING BELOW HERE”.

Notes:

- Remember to use double backslashes “\\” to allow the script to run successfully.
- After you fill the parameters, it should look like this:

*Figure 14 mkfastafile.R script after adding details.*

- Press CTRL+A to select the entire code

The screenshot shows an RStudio interface with several tabs at the top: 'File', 'Edit', 'Code', 'View', 'Plots', 'Session', 'Build', 'Debug', 'Profile', 'Tools', and 'Help'. Below the tabs, there's a search bar labeled 'Go to file/function' and an 'Addins' dropdown. The main area is a code editor with the following content:

```
1 #MK FASTA
2 #Create FASTA FILES and FILE WITH UNIQUE ID for BLAST.
3 #PLEASE FILL THE FOLLOWING
4
5 python_exe <- "C:\\\\Users\\\\s433088\\\\AppData\\\\Local\\\\Programs\\\\Python\\\\Python38\\\\python.exe"
6 inputCSV_file_path <- "C:\\\\thingslap\\\\Pipeline\\\\LINUX_Test\\\\SampleData_Enterococcus_faecium.csv"
7 output_path <- "C:\\\\thingslap\\\\Pipeline\\\\LINUX_Test\\\\"
8 output_name <- "SampleData_Enterococcus_faecium_R"
9 row_where_header_starts <- "6"
10 row_where_data_starts <- "7"
11 sequence_column <- "4"
12
13 - ##### DO NOT MODIFY ANYTHING BELOW HERE#####
14 - #####DO NOT MODIFY ANYTHING BELOW HERE#####
15 - #####DO NOT MODIFY ANYTHING BELOW HERE#####
52:1
```

*Figure 15 mkfastafile.R script after selecting all the code.*

- Run the code

The screenshot shows the RStudio interface for the omicR package. The top menu bar includes File, Edit, Code, View, Plots, Session, Build, Debug, Profile, Tools, Help, and Addins. The left sidebar has tabs for Console, Jobs, and Pipeline. The main workspace shows a script editor with several open files and a command line interface (CLI) at the bottom. The right sidebar displays the Global Environment and Values panes.

**Code Editor:**

```
1 #MK FASTA
2 #Create FASTA FILES and FILE WITH UNIQUE ID FOR BLAST.
3 #PLEASE FILL THE FOLLOWING
4
5 python_exe <- "C:\\\\Users\\\\s433088\\\\AppData\\\\Local\\\\Programs\\\\Python\\\\Python38\\\\"
6 inputCSV_file_path <- "c:\\\\thingslap\\\\Pipeline\\\\LINUX_Test\\\\SampleData_Enterococcus_Faecium_R.csv"
7 output_path <- "c:\\\\thingslap\\\\Pipeline\\\\LINUX_Test\\\\"
8 output_name <- "SampleData_Enterococcus_faecium_R"
9 row_where_header_starts <- "6"
10 row_where_data_starts <- "7"
11 sequence_column <- "4"
12
13 ##### DO NOT MODIFY ANYTHING BELOW HERE#####
14 #####
15 #####
8:51 (Top Level) R Script
```

**Environment pane (Global Environment):**

Name	Type	Value
input...	character	"C:\\\\thingslap...
output...	character	"SampleData_En...
outputp...	character	"C:\\\\thingslap...
pytho...	character	"C:\\\\Users\\\\s4...
pytho...	character	"C:\\\\Users\\\\s4...
row_W...	character	"7"
row_W...	character	"6"
seque...	character	"4"

**Console pane:**

```
C:\\thingslap\\Pipeline\\RPackage\\omicR>
+ "-f", sequence_column
+ )
> system(python_command_line_TestingPyCharm_Mkfasta)
[1] 0
>
```

*Figure 16 mkfastafolder.R script after running the code.*

This code will produce two files that are needed for the following steps.

**Fasta file**

A
1 >1
2 TGCGAAGAAGTACGAAGAGAACAGAACTCTACGCCTGAAACACCG
3 >2
4 TGCGGGGCCATCATACGGGATAACGACTGTATGACGTGAAACCG
5 >3
6 TGCACTACGGAATCTTCGATTATCAGGAAGTCGAGCCG
7 >4
8 TGCACTGCTGTTCTGGCACCATATTCGCGAAGTCGG
9 >5
10 TGCACTGCATTGGCTTCGATTACTTGATGCAAGAACATCCG
11 >6
12 TGCACTCGCTTGAAGAACTAGCGGTTACGTATTAGAACCG
13 >7
14 TGCACTGATATCCGTTACCTAGCTGAAACGCTAGAGAACAAACCG
15 >8
16 TGCAACGCATCATATATTGGCTTAACGATTGTGCCCC
17 >9
18 TGCACTCTGGTAAATTCTCTAGCATCAACCAAGAACGCTACAAACCG
19 >10
20 TGCACTGCTTGTGAGTTACAGAACCGGAGAACGCTACAAACCG

**File with FastaFileID**

A	B	C	D	E
1 FastaFileID	SeqIndex	ClusterIdx	ClusterSiz/Tag	TrimmedSequence
2 1	89184	20963	408	TGCGAAGAAGTACGAAGAGAACACCG
3 2	75556	20878	111	TGCGGGGCCATCATACGGGAT
4 3	73901	20862	96	TGCACTACGGAATCTTCGATT
5 4	79622	20911	137	TGCACTGCTGTTCTGGCACCA
6 5	78436	20902	127	TGCACTGATATCCGTTACGT
7 6	78175	20900	126	TGCAACGCATCATATATTGG
8 7	79343	20909	134	TGCACTGCTGTTCTGGTAAAT
9 8	76464	20886	118	TGCAACGCTAGAACGCTAC
10 9	79759	20912	137	TGCACTGCTGTTCTGGTAAAT
11 10	80025	20914	139	TGCACTGCTGTTCTGGTAAAT
12 11	79492	20910	136	TGCACTGCTGTTCTGGTAAAT
13 12	80161	20915	141	TGCACTGCTGTTCTGGTAAAT
14 13	76107	20883	117	TGCACTGCTGTTCTGGTAAAT
15 14	82781	20933	156	TGCACTGCTGTTCTGGTAAAT
16 15	75759	20880	112	TGCACTGCTGTTCTGGTAAAT
17 16	85903	20950	214	TGCACTGCTGTTCTGGTAAAT
18 17	83279	20936	163	TGCACTGCTGTTCTGGTAAAT
19 18	83781	20939	179	TGCACTGCTGTTCTGGTAAAT
20 19	83415	20937	171	TGCACTGCTGTTCTGGTAAAT

Figure 17 Example of fasta file and file with Unique ID.

## Download genomes from the NCBI.

- Select the script “downloadGenomes.R”.
- Clean your global environment before running this script.

```

# DOWNLOAD GENOMES
#Download genomes from NCBI using accession number.
#PLEASE FILL THE FOLLOWING
python_exe <- ""
email <- ""
genomeAccessions <- ""
OutputFilePath <- ""
fileName <- ""

#### DO NOT MODIFY ANYTHING BELOW HERE#####

```

Figure 18 Example of downloadGenomes.R script.

- Fill the scripts with the path to the python executable files, your email, **the RefSeq numbers separated by a comma and WITHOUT SPACE IN BETWEEN**, the output file path and name.

➤ Remember to use \\ in all your paths.

The screenshot shows the RStudio interface with the 'omicR - RStudio' window open. The code editor contains the following R script:

```

1 #DOWNLOAD GENOMES
2
3 #Download genomes from NCBI using accession number.
4 #PLEASE FILL THE FOLLOWING
5
6 python_exe <- "C:\\Users\\s433088\\AppData\\Local\\Programs\\Python\\Python38\\python.exe"
7 email <- "berenicetalamantes@yahoo.fr"
8 genomeAccessions <- "NZ_CP039729.1,NZ_CP039730.1"
9 outputPath <- "C:\\thingslap\\Pipeline\\LINUX_Test\\"
10 fileName <- "E_faecium_genome"
11
12
13 ##### DO NOT MODIFY ANYTHING BELOW HERE#####
14 #####
15

```

The 'Environment' pane on the right shows the variables defined in the script:

- email: "berenicetalamantes@yahoo.fr"
- fileName: "E\_faecium\_genome"
- genomeAccessions: "NZ\_CP039729.1,NZ\_CP039730.1"
- outputPath: "C:\\thingslap\\Pipeline\\LINUX\_Test\\"
- python\_exe: "C:\\Users\\s433088\\AppData\\Local\\Programs\\Python\\Python38\\python.exe"
- python...: "C:\\\\Users\\\\s433088\\\\AppData\\\\Local\\\\Programs\\\\Python\\\\Python38\\\\python.exe"

Figure 19 Example of downloadGenomes.R script after adding parameters.

- Save your script, select all the code and run.

The screenshot shows the RStudio interface with the 'omicR - RStudio' window open. The code editor contains the same R script as Figure 19. The 'Console' pane at the bottom shows the following output:

```

> system(python_command_line_TestingPyCharm_Downloading_genomes)
['NZ_CP039729.1', 'NZ_CP039730.1']
[>NZ_CP039729.1 Enterococcus faecium strain ZY2 chromosome, complete genome]
1
['>NZ_CP039730.1 Enterococcus faecium strain ZY2 plasmid pZY2']
2
[1] 0
>

```

The 'Environment' pane on the right shows the variables defined in the script:

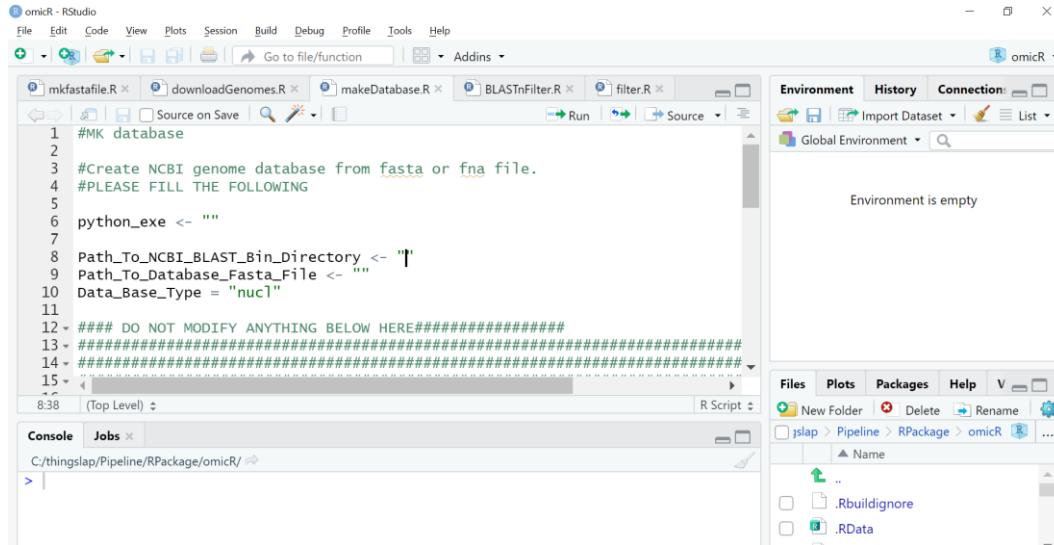
- email: "berenicetalamantes@yahoo.fr"
- fileName: "E\_faecium\_genome"
- genomeAccessions: "NZ\_CP039729.1,NZ\_CP039730.1"
- outputPath: "C:\\thingslap\\Pipeline\\LINUX\_Test\\"
- python\_exe: "C:\\Users\\s433088\\AppData\\Local\\Programs\\Python\\Python38\\python.exe"
- python...: "C:\\\\Users\\\\s433088\\\\AppData\\\\Local\\\\Programs\\\\Python\\\\Python38\\\\python.exe"

Figure 20 Example of downloadGenomes.R script after running the code.

This step creates a directory with a single fasta file that includes all the RefSeq numbers fetched.

## Create BLAST+ database.

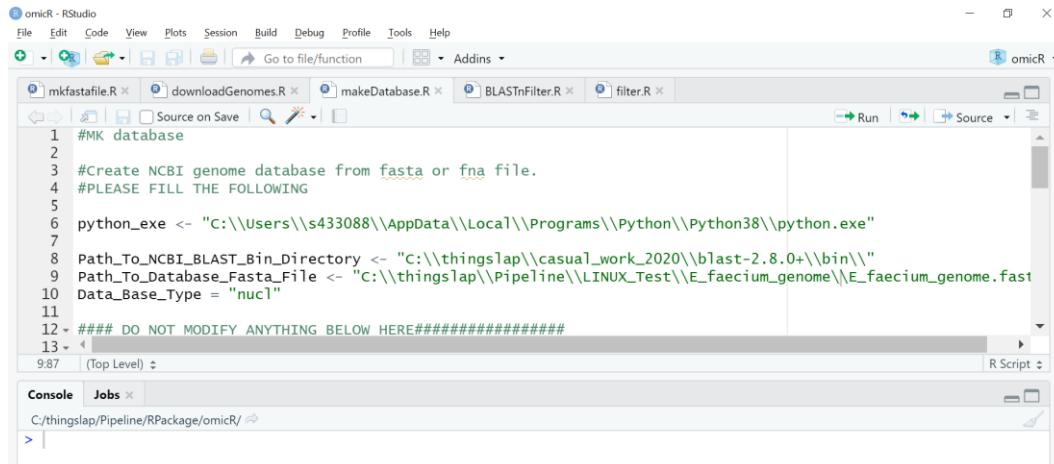
- Select the script “makeDatabase.R” and complete the script by typing the paths for your computer.
  - Clean your global environment before running this script.



*Figure 21 Example of window showing “makeDatabase.R”*

- Fill the script and add the path to the Python executable, path to the BLAST+ bin directory, path to the file (genome, contigs, scaffolds...etc) to turn into database, and the type of database (for nucleotides select “nucl”).

- Remember to type \\ in the path.



*Figure 22 Example of script “makeDatabase.R” after filling parameters.*

- Save the script, select all the code and run it.

```

1 #MK database
2
3 #Create NCBI genome database from fasta or fna file.
4 #PLEASE FILL THE FOLLOWING
5
6 python_exe <- "C:\\Users\\s433088\\AppData\\Local\\Programs\\Python\\Python38\\python.exe"
7
8 Path_To_NCBI_BLAST_Bin_Directory <- "c:\\thingslap\\casual_work_2020\\blast-2.8.0+\\bin\\"
9 Path_To_Database_Fasta_File <- "C:\\thingslap\\Pipeline\\LINUX_Test\\E_faecium_genome\\E_faecium_genome.fasta"
10 Data_Base_Type = "nuc1"
11
12 (Untitled) 48:1

```

Console output:

```

Building a new DB, current time: 08/25/2020 12:08:37
New DB name: C:\\thingslap\\Pipeline\\LINUX_Test\\E_faecium_genome\\E_faecium_genome.fasta
New DB title: C:\\thingslap\\Pipeline\\LINUX_Test\\E_faecium_genome\\E_faecium_genome.fasta
Sequence type: Nucleotide
Deleted existing Nucleotide BLAST database named C:\\thingslap\\Pipeline\\LINUX_Test\\E_faecium_genome\\E_faecium_genome.fasta
Keep MBits: T
Maximum file size: 1000000000B
Adding sequences from FASTA; added 2 sequences in 0.0285121 seconds.

```

Figure 23 Example of script “makeDatabase.R” after running the code.

- This script produces 3 additional files with the same name as the file used to create the database. These files have the terminations: “nhr”, “nin” and “nsq”.

Name	Date modified	Type	Size
E_faecium_genome.fasta	25/08/2020 11:57 AM	FASTA File	2,769 KB
E_faecium_genome.fasta.nhr	25/08/2020 12:08 PM	NHR File	1 KB
E_faecium_genome.fasta.nin	25/08/2020 12:08 PM	NIN File	1 KB
E_faecium_genome.fasta.nsq	25/08/2020 12:08 PM	NSQ File	692 KB

Figure 24 Example of files created with the code “makeDatabase.R”.

## BLAST and filtering.

- Select the script “BLASTnFilter.R”.
- Clean your global environment before running this script.

```

1 #This script BLAST nucleotides sequences and filters
2 #
3 #PLEASE FILL THE FOLLOWING
4
5 python_exe <- ""
6 Path_to_NCBI_Directory <- ""
7 DC_MegaBlast_BF <- "FALSE"
8 Data_Base <- "C:\\thingslap\\Pipeline\\LINUX_Test\\E_faecium_genome\\E_faecium_genome.fasta"
9 Query_fasta_file <- "C:\\thingslap\\Pipeline\\LINUX_Test\\SampleData_Enterococcus_faecium_R.fasta"
10 Output_Path_ <- "C:\\thingslap\\Pipeline\\LINUX_Test\\"
11 #BLAST PARAMETRES
12 output_file_name <- "Enterococcus_faecium_R_BLAST_"
13 word_size <- "11"
14 Percentage_identity <- "70"
15 number_of_threads <- "4"
16 OutputFormat <- "6"
17 Percentage_overlap <- "0.8"
18 bitscore <- "50"
19 InputFile_with_unique_ID <- "C:\\thingslap\\Pipeline\\LINUX_Test\\SampleData_Enterococcus_faecium_Inpu
20
21 -> "
22

```

Figure 25 Example of window showing BLASTnFilter.R script.

- Fill the script and add the path to the Python executable, path to the BLAST+ bin directory, write TRUE if you are running a Discontinuous mega BLAST or FALSE if you are not, path to the BLAST+ database, path to the fasta file query, output path, output name, word size, percentage identity, number of threads, output format (only tabular format 6 is accepted), percentage overlap, bitscore, and path to file with Unique ID (Created in step 1).

```

1 #This script BLAST nucleotides sequences and filters
2 #
3 #PLEASE FILL THE FOLLOWING
4
5 python_exe <- "C:\\Users\\s433088\\AppData\\Local\\Programs\\Python\\Python38\\python.exe"
6 Path_to_NCBI_Directory <- "C:\\thingslap\\casual_work_2020\\blast-2.8.0+\\bin\\"
7 DC_MegaBlast_BF <- "FALSE"
8 Data_Base <- "C:\\thingslap\\Pipeline\\LINUX_Test\\E_faecium_genome\\E_faecium_genome.fasta"
9 Query_fasta_file <- "C:\\thingslap\\Pipeline\\LINUX_Test\\SampleData_Enterococcus_faecium_R.fasta"
10 Output_Path_ <- "C:\\thingslap\\Pipeline\\LINUX_Test\\"
11 #BLAST PARAMETRES
12 output_file_name <- "Enterococcus_faecium_R_BLAST_"
13 word_size <- "11"
14 Percentage_identity <- "70"
15 number_of_threads <- "4"
16 OutputFormat <- "6"
17 Percentage_overlap <- "0.8"
18 bitscore <- "50"
19 InputFile_with_unique_ID <- "C:\\thingslap\\Pipeline\\LINUX_Test\\SampleData_Enterococcus_faecium_Inpu
20
21 -> "
22

```

Figure 26 Example of script BLASTnFilter.R after filling parameters.

- Save the script, select all the code with CRTL+A, and run the code.

```

1 #This script BLAST nucleotides sequences and filters
2 #
3 #PLEASE FILL THE FOLLOWING
4
5 python_exe <- "C:\\\\Users\\\\s433088\\\\AppData\\\\Local\\\\Programs\\\\Python\\\\Python38\\\\pyt"
6 Path_to_NCBI_Directory <- "C:\\\\thingslap\\\\casual_work_2020\\\\blast-2.8.0+\\\\bin\\\\"
7 DC_MegaBlast_BF <- "FALSE"
8 Data_Base <- "C:\\\\thingslap\\\\Pipeline\\\\LINUX_Test\\\\E_faecium_genome\\\\E_faecium_genome"
9 Query_fasta_file <- "C:\\\\thingslap\\\\Pipeline\\\\LINUX_Test\\\\SampleData_Enterococcus_"
10 Output_Path <- "C:\\\\thingslap\\\\Pipeline\\\\LINUX_Test\\\\"
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61

```

994 NZ\_CP039729.1 NZ\_CP039729.1 Enterococcus faecium strain ZY2 chromosome, complete genome TGCAAGTTGAGGAAGAGAAAAAGAAGGGTTTGTCACAAAAATCACCG 92.000 50 3.14e-14  
AAAAAAAGAAGGGTTGGTCAACAAAATTACCG 46 4 71.3 1 50 1986419 1986468 0 0 50 2736723 1.0

[1] 0  
> |

Figure 27 Example of script BLASTnFilter.R after running the code.

This script creates 5 files.

Name	Date modified	Type	Size
Enterococcus faecium_R_BLAST_all_sequences_with_and_without_hits.txt	25/08/2020 12:47 PM	Text Document	425 KB
Enterococcus faecium_R_BLAST_filtered.txt	25/08/2020 12:47 PM	Text Document	257 KB
Enterococcus faecium_R_BLAST_only_sequences_with_hits.txt	25/08/2020 12:47 PM	Text Document	377 KB
Enterococcus faecium_R_BLAST_sorted.txt	25/08/2020 12:47 PM	Text Document	215 KB
Enterococcus faecium_R_BLAST_BLAST.txt	25/08/2020 12:47 PM	Text Document	3,057 KB

Figure 28 Example of files created by BLASTnFilter.R script.

This step takes less than 5 minutes. The script produces 5 files if you included all inputs, or 3 files if you did not provide the File with UniqueID.

### Description of files produced.

- **File 1. Enterococcus\_faecium\_R\_BLAST\_BLAST.txt**

This is the raw BLAST output. This file does not contain any headers and it is not filtered.

- **File 2. Enterococcus\_faecium\_R\_BLAST\_\_filtered.txt**

This file has headers and an extra column with the percentage overlap. Default filtering parameters for this tutorial are: Percentage Overlap >80%, bitscore >50, Percentage Identity>70. The percentage overlap can be modified according to the BLAST results expected. This file may contain multiple hits per sequence.

The BLAST output is formatted as a table using output format 6, with columns defined in the following order: " qseqid sacc stitle qseq sseq nident mismatch pident length eval evalue bitscore qstart qend sstart send gapopen gaps qlen slen". These are:

qseqid: query (e.g., unknown gene) sequence id  
sacc: Subject accession  
stitle: Subject Title  
qseq: Aligned part of query sequence  
sseq: Aligned part of subject sequence  
nident: Number of identical matches  
mismatch: number of mismatches  
pident: percentage of identical matches  
length: alignment length (sequence overlap)  
evalue: expect value  
bitscore: bit score  
qstart: start of alignment in query  
qend: end of alignment in query  
sstart: start of alignment in subject  
send: end of alignment in subject  
gapopen: number of gap openings  
gaps: Total number of gaps  
qlen: Query sequence length  
slen: Subject sequence length

- **File 3. Enterococcus\_faecium\_R\_BLAST\_\_sorted.txt**

This file contains only one hit per sequence. The best match will be selected by considering the following values ranked in order. First considering the highest percentage identity, then the highest Percentage overlap, then the highest bitscore. Only one Query per sequence is kept based on these selection criteria.

If you did not provide the file with UniqueID, this filtered and sorted BLAST output file will be your final results.

- **File 4. Enterococcus\_faecium\_R\_BLAST\_\_only\_sequences\_with\_hits.txt**

This file uses the UniqueID assigned to each sequence and writes the BLAST results back into the original file. This file only contains sequences that had a BLAST hit to something in the reference.

- **File 5. Enterococcus\_faecium\_R\_BLAST\_\_all\_sequences\_with\_and\_without\_hits.txt**

This file uses the UniqueID assigned to each sequence and writes the BLAST results back into the original file. This file contains all sequences, including those with and without hits, written back into the original file.

## Running BLAST and filtering script without the file with Unique ID.

If you do not have the file with Unique ID created with this pipeline, you can still run the BLAST and filtering. The steps are the same for running a normal BLAST but leave empty the string “InputFile\_with\_unique\_ID” as shown in the picture.

```

#BLAST PARAMETERS
Output_file_name <- "Enterococcus_faecium_R_BLAST_WIwithout"
word_size <- "11"
Percentage_identity <- "70"
number_of_threads <- "4"
OutputFormat <- "6"
Percentage_overlap <- "0.8"
bitscore <- "50"
InputFile_with_unique_ID <- ""

```

Traceback (most recent call last):  
 File "TestingPyCharm\_BLAST\_filtering\_and\_all.py", line 493, in <module>  
 main(sys.argv[1:])  
 File "TestingPyCharm\_BLAST\_filtering\_and\_all.py", line 442, in main  
 original\_Modified\_file\_BF = open(Output\_extra\_file\_BF, 'r')  
 FileNotFoundError: [Errno 2] No such file or directory:  
 [1] 1
 > |

Figure 29 Example of script BLASTnFilter.R after running the code without the file with UniqueID.

Running the code without this file will produce 4 text documents. Three of them with data and the final one empty. Your final file with the sequences of interest will have the termination “\_sorted.txt”.

Name	Date modified	Type	Size
Enterococcus_faecium_R_BLAST_WIwithout_all_sequences_with_and_without_hits.txt	25/08/2020 2:16 PM	Text Document	0 KB
Enterococcus_faecium_R_BLAST_WIwithout_sorted.txt	25/08/2020 2:16 PM	Text Document	215 KB
Enterococcus_faecium_R_BLAST_WIwithout_filtered.txt	25/08/2020 2:16 PM	Text Document	257 KB
Enterococcus_faecium_R_BLAST_WIwithoutBLAST.txt	25/08/2020 2:16 PM	Text Document	3,057 KB

Figure 30 Example output files generated after running the script BLASTnFilter.R without the file with UniqueID.

## Additional filtering.

- For further filtering with different parameters without running BLAST+ again, you can use the script “filter.R”.
- Clean the global environment before you run this script.

```

1 #This script filters a BLAST file
2 #
3 #PLEASE FILL THE FOLLOWING
4
5 python_exe <- ""
6 a_BLAST_input_path_and_file_ <- ""
7 b_output_Path_ <- ""
8 c_output_file_name <- ""
9 d_Percentage_overlap <- ""
10 e_bitscore <- ""
11
12 ##### DO NOT MODIFY ANYTHING BELOW HERE#####
13 #####
14 #####
15 #####
16 #####
17 #####

```

Figure 31 Example of filter.R script.

- Fill the script and add the path to the Python executable, path to the BLAST output file produced in the previous step, output path, output name, percentage overlap and bitscore.
- This script can work on any BLAST output file which has exactly the same format as is used here. The format used here is BLAST output format 6, with the following columns: qseqid sacc stitle qseq sseq nident mismatch pident length evalue bitscore qstart qend sstart send gapopen gaps glen slen.

```

1 #This script filters a BLAST file
2 #
3 #PLEASE FILL THE FOLLOWING
4
5 python_exe <- "C:\\Users\\s433088\\AppData\\Local\\Programs\\Python\\Python38\\python.exe"
6 a_BLAST_input_path_and_file_ <- "C:\\thingslap\\Pipeline\\LINUX_Test\\Enterococcus_faecium_R_BLAST_BLAST.txt"
7 b_output_Path_ <- "C:\\thingslap\\Pipeline\\LINUX_Test\\"
8 c_output_file_name <- "New_filter"
9 d_Percentage_overlap <- "0.01"
10 e_bitscore <- "30"
11
12 ##### DO NOT MODIFY ANYTHING BELOW HERE#####
13 #####
14 #####
15 #####
16 #####
17 #####
18 #####
19 #####
20 #####

```

Figure 32 Example of "filter.R script" after filling parameters.

- Save the script, select all the code with CRTL+A, and run the code.

```

1 #This script filters a BLAST file
2 #
3 #PLEASE FILL THE FOLLOWING
4
5 python_exe <- "C:\\Users\\s433088\\AppData\\Local\\Programs\\Python\\Python38\\"
6 a_BLAST_input_path_and_file_ <- "C:\\thingslap\\Pipeline\\LINUX_Test\\Enterococcus_faecium.strain.ZY2_chromosome.complete_genome"
7 b_output_Path_ <- "C:\\thingslap\\Pipeline\\LINUX_Test\\"
8 c_output_file_name <- "New_filter"
9 d_Percentage_overlap <- "0.01"
10 e_bitscore <- "30"
11

```

key already in dictionary  
[1] 0  
> |

Figure 33 Example of "filter.R script" after running the code.

This will produce 2 files, with the new filtering parameters selected. The file with the name you provided and the ending “\_sorted.txt” document is your final file.

New_filter_filtered.txt	25/08/2020 2:42 PM	Text Document	424 KB
New_filter_sorted.txt	25/08/2020 2:42 PM	Text Document	240 KB

## Common errors

- If R is running, not producing any script and giving this error [1] 127, it means that the compiler is not found. Check that you are selecting the correct path to the Python executable and writing \\ in the path.

```

> system(python_command_line_TestingPyCharm_Mkfasta)
[1] 127
>

```

- Writing paths with a blank space between words.
- Selecting the wrong file or wrong path.
- Selecting “excel files” instead of “csv” files as input for initial step.
- Not adding “\\” at the end of each path to a directory.

# omicR for HPC computers

## Requirements

- NCBI BLAST+ V4 or latest. (<https://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/>)
- Python V3 or latest (<https://www.python.org/downloads/>)
- Biopython (<https://biopython.org/>)
- omicR ([https://github.com/BTalamantesBecerra/omicR\\_linux](https://github.com/BTalamantesBecerra/omicR_linux))

If you are using Windows, I recommend downloading and installing Putty and WinSCP.

Add these programs to your environment path variables.

## Introduction

If you are running omicR with an HPC computer, it likely that you know how to use a command line. For this purpose, I suggest that you only use 2 scripts to “create fasta files” and “filter”. As the steps of downloading, creating a database and running BLAST can take longer than running BLAST+ directly.

The required input BLAST command line to run this filtering script is:

```
blastn -db [ ] -query [ ] -out [ ] -word_size [ ] -perc_identity [ ] -num_threads [ ] -outfmt ' 6 qseqid sacc  
stitle qseq sseq nident mismatch pident length evalue bitscore qstart qend sstart send gapopen gaps  
qlen slen'
```

In the following section, I will describe the steps for running all scripts.

# STEPS

## 1. Create FASTA files and input files for BLAST / filtering.

Structure of the command line.

```
python3 (Call Python)  
TestingPyCharm_Mkfasta.py (Name of the Python script)  
-a (Path and name of Input CSV File)  
-b (Output path)  
-c (Name of output file)  
-d (Row with header- start counting from 1)  
-e (Row with data- start counting from 1)  
-f (Column with sequence of interest - start counting from 0)
```

Sequence of interest: Column 4

	A	B	C	D	E	F	G	H	I	J	K
1	*	*	*	*	*	*	*	*	*	*	*
2	*	*	*	*	*	*	*	*	*	*	*
3	*	*	*	*	*	*	*	*	*	*	*
4	*	*	*	*	*	*	*	*	*	*	*
5	*	*	*	*	*	*	*	*	*	*	*
6	SeqIndex	ClusterIdx	ClusterSize	Tag	TrimmedSLength	LowComlChrom	ChromPos	ChromPos	AlnCnt	AlnValue	
7	89184	20963	408	TGCAGAA	TGCAGAA	49	0	0	0	999	
8	75556	20878	111	TGCAGCG	TGCAGCG	46	0	0	0	999	
9	73901	20862	96	TGCAGTA	TGCAGTA	41	0	0	0	999	
10	79622	20911	137	TGCAGTT	TGCAGTT	40	0	0	0	999	
11	78436	20902	127	TGCAGCT	TGCAGCT	43	0	0	0	999	
12	78175	20900	126	TGCAGCA	TGCAGCA	46	0	0	0	999	
13	79343	20909	134	TGCAGAT	TGCAGAT	52	0	0	0	999	

Figure 34 Display of how rows and headers should be selected.

An example of how the command line should look is below:

```
Python3 TestingPyCharm_Mkfasta.py -a ~Path\YourFile.csv -b
~Path\Directory\ -c YourFileName -d 7 -e 8 -f 2
```

**Note: Before running this command line, you must be in the location where the script is saved to be able to run it.**

**Move directories until you get to the location where the Python script was saved. If you are in Dungog, you should move to: /data/scratch/test\_talamantes/Python\_tutorial/**

If you are in the University of Canberra Network and you have access to Dungog, and nobody has deleted my directory in the "scratch" section, you can copy, paste, and run this command line:

```
python3 TestingPyCharm_MKfasta.py -a
/data/scratch/test_talamantes/Python_tutorial/SampleData_Enterococcus_faecium.csv -b
/data/scratch/test_talamantes/Python_tutorial/Outputs/ -c
Enterococcus_faecium -d 6 -e 7 -f 4
```

Use this command line as template before running tests with your data.

Notes from Arthur Georges:

"To access Dungog in command-line mode, use an SSH client like Putty from Microsoft Windows or the built-in terminal on the Mac. The host name is **dungog.win.canberra.edu.au**, port 22, Connection Type SSH. Should this not work, try host name 137.92.99.179, port 22 and type SSH. Type your username and password"

It looks like this in Dungong

```
[talamantes@dungog:/data/scratch/test_talamantes/Python_tutorial]$ python3 TestingPyCharm_MKfasta.py -a /data/scratch/test_talamantes/Python_tutorial/SampleData_Enterococcus_faecium.csv -b /data/scratch/test_talamantes/Python_tutorial/Outputs/ -c Enterococcus_faecium -d 6 -e 7 -f 4
[talamantes@dungog Python_tutorial]$
```

The two output files should be in the Outputs directory:

/data/scratch/test\_talamantes/Python\_tutorial/Outputs/

```
[talamantes@dungog Outputs]$ ls
Enterococcus_faecium.fasta  Enterococcus_faecium_InputFile_with_unique_ID.txt
[talamantes@dungog Outputs]$
```

## 2. Download genomes from the NCBI.

Structure of the command line.

**python3** (Call Python)

**TestingPyCharm\_Downloading\_genomes.py** (Name of the Python script)

**-a** (your email)

**-b** (RefSeq number)

**-c** (Path for outputs)

**-d** (Name of output file)

An example of how the command line should look is given below:

```
Python3 TestingPyCharm_Downloading_genomes.py -a
youremail@hotmail.com -b GCF_900067755.1 -c ~Path\Directory\ -d
YourGenomeName
```

This script is recommended for small genomes or chromosomes. If you are planning to download large genomes from NCBI, it is recommended to use the browser to download from the NCBI website. You cannot modify the speed of fetching in this step. For example, a bacterial genome such as NZ\_CP039730.1 takes a few seconds, but the entire chicken genome can take approximately 5 hours.

You need to provide your email as NCBI needs to identify you for granting you access, otherwise your access can be denied.

**Note: Before running this command line, you must be in the location where the script is saved to be able to run it.**

**Move directories until you get to the location where the Python script was saved. If you are in Dungog, you should move to: /data/scratch/test\_talamantes/PythonTutorial/**

If you are in the University of Canberra Network and you have access to Dungog, and nobody has deleted my directory in the "scratch" section, you can copy, paste, and run this command line:

```
python3 TestingPyCharm_Downloading_genomes.py -a newton685@hotmail.com -b  
"NZ_CP039729.1, NZ_CP039730.1" -c  
/data/scratch/test_talamantes/PythonTutorial/Outputs/ -d E_faecium_genome
```

**WARNING: THIS OPTION IS NOT WORKING UNTIL BIOPYTHON IS INSTALLED IN DUNGOG.**

### 3. Create NCBI database for BLAST+

Structure of the command line.

**python3** (Call Python)

**TestingPyCharm\_MakeDataBase.py** (Name of the Python script)

**-a** Path to your BLAST+ bin directory

**-b** (Path to your genome file as fasta or fna file)

**-c** (Type of database, use nucl for nucleotides)

An example of how the command line should look is given below:

```
python3 TestingPyCharm_MakeDataBase.py -a ~PATH\blast-2.8.0+\bin\ -b ~Path\MyGenome.fna -c  
nucl
```

#### Notes:

\* Before running this command line, you must be in the location where the script is saved to be able to run it.

\* Move directories until you get to the location where the Python script was saved. If you are in Dungog, you should move to: /data/scratch/test\_talamantes/Python\_tutorial/

\* If you already have the mkblastdb in the global environment paths do not include the handle '-a'

If you are in the University of Canberra Network and you have access to Dungog, and nobody has deleted my directory in the "scratch" section, you can copy, paste, and run this command line:

```
python3 TestingPyCharm_MakeDataBase.py -b  
/data/scratch/test_talamantes/Python_tutorial/Outputs/E_faecium_genome/E  
_faecium_genome.fasta -c nucl
```

Use this command line as template before running tests with your data.

It looks like this in Dungong

If it worked, it looks like this

The outputs are here →

```

Building a new DB, current time: 08/25/2020 16:59:51
New DB name: /data/scratch/test_talamantes/Python_tutorial/Outputs/E_faecium_genome/E_faecium_genome.fasta
New DB title: /data/scratch/test_talamantes/Python_tutorial/Outputs/E_faecium_genome/E_faecium_genome.fasta
Sequence type: Nucleotide
Keep MBits: T
Maximum file size: 1000000000B
Adding sequences from FASTA; added 2 sequences in 0.029314 seconds.
[talamantes@dungog Python_tutorial]$ dir
Outputs          TestingPyCharm_BLAST_filtering_and_all.py  TestingPyCharm_MakeDataBase.py  TestingPyCharm_NCBI_BLAST_filtering.py
SampleData_Enterococcus_faecium.csv  TestingPyCharm_Downloading_genomes.py  TestingPyCharm_MKfasta.py
[talamantes@dungog Python_tutorial]$ cd Outputs/
[talamantes@dungog Outputs]$ ls
E_faecium_genome  Enterococcus_faecium.fasta  Enterococcus_faecium_InputFile_with_unique_ID.txt
[talamantes@dungog Outputs]$ cd E_faecium_genome/
[talamantes@dungog E_faecium_genome]$ ls
E_faecium_genome.fasta  E_faecium_genome.fasta.nhr  E_faecium_genome.fasta.nin  E_faecium_genome.fasta.nsq
[talamantes@dungog E_faecium_genome]$ █

```

#### 4. BLAST and filtering.

Structure of the command line:

**python3** (Call Python)

**TestingPyCharm\_BLAST\_filtering\_and\_all.py** (Name of the Python script)

**-y** (TRUE if you are running a Dissimilar discontinuous MegaBLAST or FALSE if you are not)

**-a** (Path to the reference genome)

**-b** (Path to the query sequences)

**-c** (Path to the output directory)

**-d** (Name of output directory)

**-e** (Word Size: recommended value 11)

**-f** (Percentage identity: recommended value 70)

**-g** (Threads)

**-i** (Output format, only tabular format 6 is accepted for this script)

**-j** (Percentage overlap: recommended value 0.8)

**-k** (Bitscore: recommended value 50)

An example of how the command line should look like is below:

```
python3 TestingPyCharm_BLAST_filtering_and_all.py -y FALSE -a
Path/MyGenome.fna -b Path/MySequences.fasta -c ~Path/directory/ -d
MyBlastResults -e 11 -f 70 -g 35 -i 6 -j 0.8 -k 50 -l
```

**Notes: Before running this command line, you must be in the location where the script is saved to be able to run it.**

**\*Move directories until you get to the location where the Python script was saved. If you are in Dungog, you should move to: /data/scratch/test\_talamantes/PythonTutorial/**

**If you are in the University of Canberra Network and you have access to Dungog, and nobody has deleted my directory in the "scratch" section, you can copy, paste, and run this command line:**

```
python3 TestingPyCharm_BLAST_filtering_and_all.py -y FALSE -a  
/data/scratch/test_talamantes/PythonTutorial/Outputs/E_faecium_genome/E  
_faecium_genome.fasta -b  
/data/scratch/test_talamantes/PythonTutorial/Outputs/Enterococcus_faeci  
um.fasta -c /data/scratch/test_talamantes/PythonTutorial/Outputs/ -d  
Enterococcus_faecium_-e 11 -f 70 -g 35 -i 6 -j 0.8 -k 50 -l  
/data/scratch/test_talamantes/PythonTutorial/Outputs/Enterococcus_faeci  
um_InputFile_with_unique_ID.txt
```

**If you do not have the file with Unique Id, your command line will look like this:**

```
python3 TestingPyCharm_BLAST_filtering_and_all.py -y FALSE -a  
/data/scratch/test_talamantes/PythonTutorial/Outputs/E_faecium_genome/E  
_faecium_genome.fasta -b  
/data/scratch/test_talamantes/PythonTutorial/Outputs/Enterococcus_faeci  
um.fasta -c /data/scratch/test_talamantes/PythonTutorial/Outputs/ -d  
Enterococcus_faecium_Results -e 11 -f 70 -g 35 -i 6 -j 0.8 -k 50
```

**Use this command line as template before running tests with your data.**

If you ran the command line without including the file with unique ID, one file without data will be generated “Enterococcus\_faecium\_Results\_all\_sequences\_with\_and\_without\_hits.txt”. This is normal. Your final file has the name “Enterococcus\_faecium\_Results\_sorted.txt”

If you ran these two command lines, your outputs will look like this in Dungong.



```

talamantes@dungog:/data/scratch/test_talamantes/Python_tutorial/Outputs
TATTACAAATGCAAAATGAAAAGCTGTCTTTGGGAATCG 45 1 97.826 46 4.56e-17 80.5 1 46 2415927 2415972 0 0 46 2
736723 1.0

987 NZ_CP039729.1 NZ_CP039729.1 Enterococcus faecium strain ZY2 chromosome, complete genome
TATTACAAATGCAAAATGAAAAGCTGTCTTTGGGAATCG 45 1 97.826 46 4.56e-17 80.5 1 46 2415927 2415972 0 0 46 2
736723 1.0

988 NZ_CP039729.1 NZ_CP039729.1 Enterococcus faecium strain ZY2 chromosome, complete genome
CATGCCTTGAAAGAACTAGGGCGTTAGCTATTAGAACCG 45 1 97.826 46 4.56e-17 80.5 1 46 2354284 2354329 0 0 46 2
736723 1.0

989 NZ_CP039729.1 NZ_CP039729.1 Enterococcus faecium strain ZY2 chromosome, complete genome
CATGCCTTGAAAGAACTAGGGCGTTAGCTATTAGAACCG 45 1 97.826 46 4.56e-17 80.5 1 46 2354284 2354329 0 0 46 2
736723 1.0

99 NZ_CP039729.1 NZ_CP039729.1 Enterococcus faecium strain ZY2 chromosome, complete genome
GCAGTTGAGGAAAAGAAAAAGAAGGGTTTGGTCACAAATAACCG 50 0 100.000 50 6.69e-21 93.5 1 50 1986419 1986468 0 0 5
0 2736723 1.0

990 NZ_CP039729.1 NZ_CP039729.1 Enterococcus faecium strain ZY2 chromosome, complete genome
CATGCCTTGAAAGAACTAGGGCGTTAGCTATTAGAACCG 45 1 97.826 46 4.56e-17 80.5 1 46 2354284 2354329 0 0 46 2
736723 1.0

991 NZ_CP039729.1 NZ_CP039729.1 Enterococcus faecium strain ZY2 chromosome, complete genome
ACAAATGTTGGACTACGTATTATGAYGT 38 3 92.683 41 6.15e-11 60.2 2 42 855172 855212 0 0 47 27367
23 0.072340425531949

992 NZ_CP039729.1 NZ_CP039729.1 Enterococcus faecium strain ZY2 chromosome, complete genome
GCAGTTGAGGAAAAGAAAAAGAAGGGTTTGGTCACAAATAACCG 46 4 92.000 50 3.14e-14 71.3 1 50 1986419 1986468 0 0 5
0 2736723 1.0

994 NZ_CP039729.1 NZ_CP039729.1 Enterococcus faecium strain ZY2 chromosome, complete genome
GCAGTTGAGGAAAAGAAAAAGAAGGGTTTGGTCACAAATAACCG 46 4 92.000 50 3.14e-14 71.3 1 50 1986419 1986468 0 0 5
0 2736723 1.0

[calamantes@dungog Python_tutorial]$ cd Outputs/
[calamantes@dungog Outputs]$ ls
E_faecium_genome Enterococcus_faecium_only_sequences_with_hits.txt
Enterococcus_faecium_all_sequences_with_and_without_hits.txt Enterococcus_faecium_Results_all_sequences_with_and_without_hits.txt
Enterococcus_faecium_BLAST.txt Enterococcus_faecium_ResultsBLAST.txt
Enterococcus_faecium_fasta Enterococcus_faecium_Results_filtered.txt
Enterococcus_faecium_filtered.txt Enterococcus_faecium_Results_sorted.txt
Enterococcus_faecium_InputFile_with_unique_ID.txt Enterococcus_faecium_sorted.txt
[calamantes@dungog Outputs]$
```

This step takes less than 5 minutes. It will produce the 5 files, or the 3 files if you did not provide the File with UniqueID.

### Description of files produced.

- File 1. Enterococcus\_faecium\_BLAST.txt**

This is the raw BLAST output. This file does not contain any headers and it is not filtered.

- File 2. Enterococcus\_faecium\_filtered.txt**

This file has headers and an extra column with the percentage overlap. Default filtering parameters for this tutorial are: Percentage Overlap >80%, bitscore >50, Percentage Identity>70. The percentage overlap can be modified according to the BLAST results expected. This file may contain multiple hits per sequence.

The BLAST output is formatted as a table using output format 6, with columns defined in the following order: " qseqid sacc stitle qseq sseq nident mismatch pident length evalue bitscore qstart qend sstart send gapopen gaps qlen slen". These are:

qseqid: query (e.g., unknown gene) sequence id

sacc: Subject accession

stitle: Subject Title

qseq: Aligned part of query sequence

sseq: Aligned part of subject sequence

nident: Number of identical matches

mismatch: number of mismatches

pident: percentage of identical matches

length: alignment length (sequence overlap)

evalue: expect value

bitscore: bit score  
qstart: start of alignment in query  
qend: end of alignment in query  
sstart: start of alignment in subject  
send: end of alignment in subject  
gapopen: number of gap openings  
gaps: Total number of gaps  
qlen: Query sequence length  
slen: Subject sequence length

- **File 3. Enterococcus\_faecium\_sorted.txt**

This file contains only one hit per sequence. The best match will be selected by considering the following values ranked in order. First considering the highest percentage identity, then the highest Percentage overlap, then the highest bitscore. Only one Query per sequence is kept based on these selection criteria.

If you did not provide the file with UniqueID, this filtered and sorted BLAST output file will be your final results.

- **File 4. Enterococcus\_faecium\_only\_sequences\_with\_hits.txt**

This file uses the UniqueID assigned to each sequence and writes the BLAST results back into the original file. This file only contains sequences that had a BLAST hit to something in the reference.

- **File 5. Enterococcus\_faecium\_all\_sequences\_with\_and\_without\_hits.txt**

This file uses the UniqueID assigned to each sequence and writes the BLAST results back into the original file. This file contains all sequences, including those with and without hits, written back into the original file.

## 5. Additional filtering.

Structure of the command line:

**python3** (Call Python)

**TestingPyCharm\_NCBI\_BLAST\_filtering.py** (Name of the Python script)

**-a** (Path and name of BLAST output file)

**-b** (Path to the output directory)

**-c** (Name of output directory)

**-d** (Percentage overlap: recommended value 0.8)

**-e** (Bitscore: recommended value 50)

An example of how the command line should look like is below:

```
python3 TestingPyCharm_NCBI_BLAST_filtering.py -a Path/MyBLAST_Results.txt  
-b ~Path/directory/ -c My_Filtered_Blast_Results -d 0.8 -e 50
```

**Note: Before running this command line, you must be in the location where the script is saved to be able to run it.**

**Move directories until you get to the location where the Python script was saved. If you are in Dungog, you should move to: /data/scratch/test\_talamantes/Python\_tutorial/**

**\*You have to run the previous step (or command lines) to be able to run this command line.**

If you are in the University of Canberra Network and you have access to Dungog, and nobody has deleted my directory in the "scratch" section, you can copy, paste, and run this command line:

```
python3 TestingPyCharm_NCBI_BLAST_filtering.py -a  
/data/scratch/test_talamantes/Python_tutorial/Outputs/Enterococcus_faecium  
_BLAST.txt -b /data/scratch/test_talamantes/Python_tutorial/Outputs/ -c  
E_faecium_new_filtering -d 0.8 -e 50
```

**Use this command line as template before running tests with your data.**

The command line in Dungong looks like this:

```
[talamantes@dungog Python_tutorial]$ python3 TestingPyCharm_NCBI_BLAST_filtering.py -a /data/scratch/test_talamantes/Python_tutorial/Outputs/Enterococcus_faecium_BLAST.txt -b /data/scratch/test_talamantes/Python_tutorial/Outputs/ -c E_faecium_new_filtering -d 0.8 -e 50
```

If you run it, the screen will look like this.

blastn -evalue 10 -db NZ_CPs -outfmt 6													
984	NZ_CP039730.1	NZ_CP039730.1	Enterococcus faecium strain ZY2 plasmid pZY2	52.8	9	45	44268	44304	0	0	46	97574	0.8043478260869565
GTGAAACC	34	3	91.892	37	9.94e-09								
985	NZ_CP039730.1	NZ_CP039730.1	Enterococcus faecium strain ZY2 plasmid pZY2	52.8	9	45	44268	44304	0	0	46	97574	0.8043478260869565
GTGAAACC	34	3	91.892	37	9.94e-09								
986	NZ_CP039729.1	NZ_CP039729.1	Enterococcus faecium strain ZY2 chromosome, complete genome	45	1	97.826	46	4.56e-17	80.5	1	46	2415927	2415972 0 0 46
TATTTACAATGACAATGGAAAGCTGTCTTGGGAATCCG													
987	NZ_CP039729.1	NZ_CP039729.1	Enterococcus faecium strain ZY2 chromosome, complete genome	45	1	97.826	46	4.56e-17	80.5	1	46	2415927	2415972 0 0 46
TATTTACAATGACAATGGAAAGCTGTCTTGGGAATCCG													
988	NZ_CP039729.1	NZ_CP039729.1	Enterococcus faecium strain ZY2 chromosome, complete genome	45	1	97.826	46	4.56e-17	80.5	1	46	2354284	2354329 0 0 46
CATCGCTTGAAGACTTAGCGGTTACGTATTAGAAGACCG													
989	NZ_CP039729.1	NZ_CP039729.1	Enterococcus faecium strain ZY2 chromosome, complete genome	45	1	97.826	46	4.56e-17	80.5	1	46	2354284	2354329 0 0 46
CATCGCTTGAAGACTTAGCGGTTACGTATTAGAAGACCG													
990	NZ_CP039729.1	NZ_CP039729.1	Enterococcus faecium strain ZY2 chromosome, complete genome	50	0	100.000	50	6.68e-21	93.5	1	50	1986419	1986468 0 0
GCAGTTGAGGAAAAGAAAAAGAAGGGTTTGTCACAAAAATTACCG													
736723	1.0												
991	NZ_CP039729.1	NZ_CP039729.1	Enterococcus faecium strain ZY2 chromosome, complete genome	38	3	92.683	41	6.15e-11	60.2	2	42	855172	855212 0 0 47
ACAAATATGTTGGATTACTGTATTATATGATGT													
0.8723404255319149													
993	NZ_CP039729.1	NZ_CP039729.1	Enterococcus faecium strain ZY2 chromosome, complete genome	46	4	92.000	50	3.14e-14	71.3	1	50	1986419	1986468 0 0
GCAGTTGAGGAAAAGAAAAAGAAGGGTTTGTCACAAAAATTACCG													
0	2736723	1.0											
994	NZ_CP039729.1	NZ_CP039729.1	Enterococcus faecium strain ZY2 chromosome, complete genome	46	4	92.000	50	3.14e-14	71.3	1	50	1986419	1986468 0 0
GCAGTTGAGGAAAAGAAAAAGAAGGGTTTGTCACAAAAATTACCG													
0	2736723	1.0											

The two new files will be in the output directory with the other files.

### Checklist for running scripts using a command line:

- Check that you are using a “csv” file as input.
- Check for blank spaces between words in your command line. Blank spaces will cause a failure. Use a text editor such as notepad or Notepad++ to build your command lines to facilitate your work.
- Check that you are in the correct location where the Python Script has been saved, each time you are running the selected script.
- Check that you have BLAST+, Biopython and Python V3 installed. Also check that they are in the System Path.

# The End